

# Assessing the Computational Complexity of Multi-Layer Subgraph Detection

Robert Bredereck<sup>1</sup>, Christian Komusiewicz<sup>2</sup>, Stefan Kratsch<sup>3</sup>, Hendrik Molter<sup>1</sup>,  
Rolf Niedermeier<sup>1</sup>, and Manuel Sorge<sup>1</sup>

<sup>1</sup>Institut für Softwaretechnik und Theoretische Informatik, TU Berlin, Germany,  
{robert.bredereck, h.molter, rolf.niedermeier, manuel.sorge}@tu-berlin.de

<sup>2</sup>Institut für Informatik, Friedrich-Schiller-Universität Jena, Germany,  
christian.komusiewicz@uni-jena.de

<sup>3</sup>Institut für Informatik I, Universität Bonn, Germany, kratsch@cs.uni-bonn.de

April 27, 2016

## Abstract

Multi-layer graphs consist of several graphs (layers) over the same vertex set. They are motivated by real-world problems where entities (vertices) are associated via multiple types of relationships (edges in different layers). We chart the border of computational (in)tractability for the class of subgraph detection problems on multi-layer graphs, including fundamental problems such as maximum matching, finding certain clique relaxations (motivated by community detection), or path problems. Mostly encountering hardness results, sometimes even for two or three layers, we can also spot some islands of tractability.

## 1 Introduction

*Multi-layer* graphs consist of several layers where the vertex set of all layers is the same but each layer has an individual edge set [4, 27, 31]. They are also known as multi-dimensional networks [3], multiplex networks [34], and edge-colored multigraphs [1, 9]. In recent years, multi-layer graphs have gained a lot of attention in the social network analysis and data mining communities because observational data often comes in a multimodal nature. Typical topics studied here include clustering [5, 14, 15, 23], detection of network communities [26, 40], data privacy [37], and general network properties [3].

In several of these applications, researchers identify vertex subsets of a multi-layer graph that exhibit a certain structure in each of the layers. For example, motivated by applications in genome comparison in computational biology, Gai et al. [21] searched for maximal vertex subsets in a two-layer graph that induce a connected graph in each of the layers. Jiang and Pei [23] and Boden et al. [5] searched for vertex subsets that induce dense subgraphs in many layers. Such vertex subsets model communities in a multimodal social network.

To the best of our knowledge, however, there is no systematic work on computational complexity classification beyond typically observing the generalization of hardness results for the one-layer case to the multi-layer one [5, 23]. Our aim in this article is hence to provide a general foundation for studying multi-layer subgraph problems, and to provide some initial results that pave the way for more specific complexity analyses.

We first give a general problem definition that encompasses the problems sketched above. Vaguely, they can be phrased as finding a large vertex subset that induces graphs with an interesting property in many layers. Motivated by the heterogeneity of the desired properties, our problem definition has as parameter a *graph property*  $\Pi$ , that is, any fixed set of graphs.

## $\Pi$ MULTI-LAYER SUBGRAPH ( $\Pi$ -ML-SUBGRAPH)

**Input:** A set of graphs  $G_1, \dots, G_t$  all on the same vertex set  $V$  and two positive integers  $k$  and  $\ell$ .

**Question:** Is there a vertex set  $X \subseteq V$  with  $|X| \geq k$  such that for at least  $\ell$  of the input graphs  $G_i$  it holds that  $G_i[X] \in \Pi$ ?

We study  $\Pi$ -ML-SUBGRAPH mostly in the context of parameterized complexity. As parameters we use the most natural candidates: the number  $t$  of layers, the order  $k$  of the desired subgraph and the number  $\ell$  of layers in which we search for our subgraph, as well as their dual deletion parameters  $|V| - k$  and  $t - \ell$ . Observe that NP-hardness and W[1]-hardness to either  $k$  or  $|V| - k$  in the single-layer case directly implies hardness of the multi-layer case.

Our analysis of  $\Pi$ -ML-SUBGRAPH starts with several easy results on hereditary graph properties  $\Pi$ , that is,  $\Pi$  is closed under taking induced subgraphs. Such properties  $\Pi$  have been well-studied in the single-layer case. Using Ramsey arguments and Khot and Raman's theorem [25], we get a trichotomy for the complexity of  $\Pi$ -ML-SUBGRAPH with respect to polynomial-time solvability and fixed-parameter tractability with respect to  $k$  and  $\ell$  (Proposition 1). Second, we generalize Cai's FPT result [8], by showing that, for graph properties  $\Pi$  characterized by a finite number of forbidden induced subgraphs,  $\Pi$ -ML-SUBGRAPH is fixed-parameter tractable with respect to  $t - \ell$  and  $|V| - k$  combined, and that it admits a polynomial-size problem kernel (Proposition 2).

Next, we turn to graph properties that are not necessarily hereditary. For finding connected graphs of order at least  $k$  in  $\ell$  of  $t$  layers, there is a simple fixed-parameter algorithm with respect to  $t$  which is also an XP-algorithm with respect to  $t - \ell$  or with respect to  $\ell$ . This algorithm admits a generalization to each graph property that implies certain good-natured partitions of the input graphs (Proposition 3), for example  $c$ -cores and  $c$ -trusses. As a counterpart, we offer a W[1]-hardness result for  $\Pi$ -ML-SUBGRAPH for the combined parameter  $k$  and  $\ell$  for a large class of graph properties  $\Pi$  that includes connected graphs,  $c$ -cores, and  $c$ -trusses, for example (Theorem 1).

Finally, we exhibit simple graph properties  $\Pi$  for which already a small number of layers lead to NP-hardness and W[1]-hardness of  $\Pi$ -ML-SUBGRAPH: While finding a vertex subset that induces subgraphs of order  $k$  with a perfect matching in two layers is polynomial-time solvable, it becomes NP-hard and W[1]-hard with respect to  $k$  in three layers (Theorem 2). Intuitively, the reason for the computational complexity transition from two layers to three layers is as follows. Overlaying two matchings one may get cycles and paths but without connections between them. We can cope with this by finding a maximum weighted matching in an auxiliary graph. Adding a third layer, however, allows arbitrary connections between cycles and paths. While finding a  $k$ -path, that is, a  $k$ -vertex graph containing a Hamiltonian path, is fixed-parameter tractable with respect to  $k$  in one layer [32], it becomes W[1]-hard in two layers (Theorem 3).

Apart from aiming to provide a broad overview over the complexity of  $\Pi$ -ML-SUBGRAPH, the main technical contributions are conditions on  $\Pi$  that make  $\Pi$ -ML-SUBGRAPH hard (Theorem 1) and understanding the transition from tractability to hardness for perfectly matchable subgraphs (Theorem 2) and Hamiltonian subgraphs (Theorem 3).

**Related Work.** As mentioned in the beginning, despite the numerous practical studies related to multi-layer networks, we are not aware of systematic work pertaining to the computational complexity of  $\Pi$ -ML-SUBGRAPH. The following special cases were studied from this viewpoint. Gai et al. [21] and [7] studied the case where  $\Pi$  contains every connected graph and  $t = \ell = 2$ . They showed that the resulting problem is polynomial-time solvable. In contrast, Cai and Ye [9] studied a modified version of this problem, where the desired vertex subset shall be of size *exactly*  $k$  instead of at least  $k$ . They showed NP-hardness and W[1]-hardness with respect to  $k$  and with respect to  $|V| - k$ . Agrawal et al. [1] gave a  $O(23^{tk} \cdot \text{poly}(n, t))$ -time algorithm for the case where  $\Pi$  contains all cycle free graphs and  $t = \ell$ .

Edge-colored graphs and multigraphs, which are equivalent to multi-layer graphs, were studied extensively. For surveys, see Bang-Jensen and Gutin [2, Chapter 16] and Kano and Li [24]. Most of the results therein, in the multi-layer terminology, pertain to paths and cycles which do not contain two consecutive edges in the same layer and to related questions like connectedness and Hamiltonicity using this notion of paths or cycles.

**Preliminaries.** We use the framework of parameterized complexity: Let  $p$  be a *parameter* for  $\Pi$ -ML-SUBGRAPH, that is, any integer depending on the input. We aim to prove *fixed-parameter tractability (FPT)* by giving an algorithm that produces a solution in  $f(p) \cdot \text{poly}(|V|, t)$  time, where  $f$  is a computable function, or we aim to show that such an algorithm is unlikely ( $W[1]$ -hardness for  $p$ ). A *problem kernel* is a model for efficient data reduction. Formally, it is a polynomial-time many-one self-reduction such that the size of each resulting instance is bounded by a function of the parameter  $p$ . For the precise definitions and methodology we refer the reader to the literature [11, 16, 20, 35].

All graphs are undirected and without self loops or multiple edges. We use standard graph notation [13]. A graph property  $\Pi$  is *hereditary* if removing any vertex from a graph in  $\Pi$  results again in a graph in  $\Pi$ . We use the following graph properties. A graph is a *c-core* if each vertex has degree at least  $c$  [38]. A graph is a *c-truss* if each edge is contained in at least  $c - 2$  triangles [10]. We say that a graph is *Hamiltonian* if it contains a simple path that comprises all vertices in the graph. A *c-factor* in a graph is a subset of the edges such that each vertex is incident with exactly  $c$  edges. In sans serif font face we often denote graph properties. For example, *c-Truss* is the set of all  $c$ -trusses. By *Matching* we refer to the set of all graphs containing a perfect matching and by *c-Factor* to the set of all graphs containing a  $c$ -factor. For further definitions of graph properties, see Appendix A.

**Organization.** In Section 2 we give general results for hereditary graph properties  $\Pi$ . In Section 3 we give general results for large classes of graph properties  $\Pi$ , that are not necessarily hereditary. In Section 4 we show that *c-Factor-ML-SUBGRAPH* is polynomial-time solvable for two layers but NP-hard and  $W[1]$ -hard with respect to  $k$  in three layers. In Section 5 we show that *Hamiltonian-ML-SUBGRAPH* is  $W[1]$ -hard with respect to  $k$  in two layers.

## 2 Hereditary Graph Properties

In this section we study the (parameterized) complexity of  $\Pi$ -ML-SUBGRAPH with respect to hereditary graph properties  $\Pi$ . We give a trichotomy with regard to polynomial-time solvability, NP-hardness, and the complexity with respect to parameters  $k$  and  $\ell$ , and we observe fixed-parameter tractability for the deletion parameters  $|V| - k$  and  $t - \ell$ .

Next, we investigate the computational complexity of  $\Pi$ -ML-SUBGRAPH for *hereditary* properties  $\Pi$ . Many natural graph properties fall into this category. The single-layer case has been studied by Lewis and Yannakakis [29] as well as Khot and Raman [25], the latter studied the parameterized complexity of the subgraph detection for hereditary properties. We generalize their results to the multi-layer case.

**Proposition 1.** *If  $\Pi$  is a hereditary graph property, then the following statements are true.*

1. *If  $\Pi$  excludes at least one complete graph and at least one edgeless graph, then  $\Pi$ -ML-SUBGRAPH is solvable in polynomial time.*
2. *If  $\Pi$  includes all complete graphs and all edgeless graphs, then  $\Pi$ -ML-SUBGRAPH is NP-hard and FPT when parameterized by  $k$  and  $\ell$  combined.*
3. *If  $\Pi$  includes either all complete graphs or all edgeless graphs, then  $\Pi$ -ML-SUBGRAPH is NP-hard and  $W[1]$ -hard when parameterized by  $k$  for all  $\ell$ .*

Note that every hereditary graph property falls into one of the three cases of Proposition 1.

*Proof.* In this proof we utilize the concept of Ramsey numbers. Herein, Ramsey number  $R(p, q)$  states the minimum number of vertices such that any graph with said number of vertices has either a clique of size  $p$  or an independent set of size  $q$ . It is well-known that  $R(p, q) \leq \binom{p+q-2}{q-1}$ . We give separate proofs for all three statements in the theorem. Note that for the second and third case, NP-hardness in the single-layer case was shown by Lewis and Yannakakis [29].

1. Let  $p, q$  be the sizes of the smallest excluded complete and edgeless graph, respectively. Note that any graph on at least  $R(p, q)$  vertices contains either a clique of size  $p$  or an independent set of size  $q$  and hence is not included in  $\Pi$ . Therefore there are only finitely many graphs that have property  $\Pi$ . Furthermore we have that if  $k \geq R(p, q)$ , we face a no-instance.

If  $k < R(p, q)$ , we can check every vertex subset  $X$  of size  $k$  on every layer and check whether  $G[X] \in \Pi$  on at least  $\ell$  layers. The time to do that is bounded by  $t \binom{n}{k} f(k)$  where  $f(k)$  the time to check membership of  $\Pi$ . Note that  $k$  is bounded by  $R(p, q)$  and  $p$  and  $q$  are constants only depending on  $\Pi$ . Hence, we get a polynomial running time.

2. For this proof, we introduce nested Ramsey numbers as follows.

$$R^{(1)}(p, q) = R(p, q),$$

$$R^{(i)}(p, q) = R(R^{(i-1)}(p, q), R^{(i-1)}(p, q)).$$

We show that if  $|V| \geq R^{(\ell)}(k, k)$  we face a yes-instance. Furthermore, we can find a vertex subset  $X$  with  $G[X] \in \Pi$  for any  $\ell$  layers. We prove this by induction on  $\ell$ .

For  $\ell = 1$  we have that  $|V| \geq R(k, k)$ , hence every layer of the graph has either a clique of size  $k$  or an independent set of size  $k$ . For  $|V| \geq R^{(\ell)}(k, k)$  we know that any layer has either a clique of size  $R^{(\ell-1)}(k, k)$  or an independent set of size  $R^{(\ell-1)}(k, k)$ . Without loss of generality let  $X' \subseteq V$  with  $|X'| = R^{(\ell-1)}(k, k)$  be either a clique or an independent set on layer 1. Consider  $G[X']$ : By the induction hypothesis we know that we can find a vertex set  $X \subseteq X'$  with  $|X| \geq k$  on any  $\ell - 1$  layers such that  $G[X] \in \Pi$ . Additionally, we have that  $G[X] \in \Pi$  on layer 1. Hence,  $G[X] \in \Pi$  on at least  $\ell$  layers.

If  $|V| < R^{(\ell)}(k, k)$ , we can find a solution by brute-force, if it exists. Note that NP-hardness follows from the NP-hardness of the single-layer case.

3. Khot and Raman [25] showed that for hereditary properties  $\Pi$  that include either all complete graphs or all edgeless graphs,  $\Pi$ -SUBGRAPH is W[1]-hard when parameterized by  $k$ . This directly translates to the multi-layer case, as does NP-hardness.  $\square$

In the following corollary, we give a number of hereditary properties  $\Pi$  and the corresponding complexity results for  $\Pi$ -ML-SUBGRAPH implied by Proposition 1. For their definitions we refer to the literature [6, 22] or to Appendix A. We remark that properties that fall into the first case are exactly those containing only a finite number of graphs.

**Corollary 1.**  $\Pi$ -ML-SUBGRAPH is NP-hard and FPT when parameterized by  $k$  and  $\ell$  combined for  $\Pi \in \{\text{Perfect Graph, Interval Graph, Chordal Graph, Split Graph, Asteroidal Triple Free Graph, Comparability Graph, Permutation Graph}\}$ .

$\Pi$ -ML-SUBGRAPH is NP-hard and W[1]-hard when parameterized by  $k$  for all  $\ell$  for  $\Pi \in \{\text{Edgeless Graph, Complete Graph, Complete Multipartite Graph, Planar Graph, } c\text{-Colorable Graph, Forest}\}$ .

Now we consider properties  $\Pi$ , whose complements are hereditary. We can show that, for them, polynomial-time solvability transfers to the multi-layer case.

**Observation 1.** Let  $\Pi$  be a graph property such that, if  $G \in \Pi$  for some graph  $G$  and  $H[X] = G$  for some graph  $H$  and vertex set  $X$ , then  $H \in \Pi$ . Equivalently, the complement property (containing all graphs not in  $\Pi$ ) is hereditary. If  $\Pi$  can be decided in  $f(n)$  time for some function  $f$ , then  $\Pi$ -ML-SUBGRAPH can be decided in  $O(t \cdot f(n))$  for all  $k$  and  $\ell$ .

*Proof.* Observe that if  $G \notin \Pi$  then no induced subgraph of  $G$  can be in  $\Pi$ . Hence to decide  $\Pi$ -ML-SUBGRAPH, we decide  $\Pi$  on each graph  $G_1, \dots, G_t$ . We face a yes-instance if and only if there are at least  $\ell$  graphs that have property  $\Pi$ : We can set  $X = V$ , and hence  $|X| \geq k$ , for any  $k \leq n$ .  $\square$

In the following corollary, we give two properties  $\Pi$  for which  $\Pi$ -ML-SUBGRAPH is solvable in polynomial time.

**Corollary 2.**  $\Pi$ -ML-SUBGRAPH is solvable in polynomial time for:

- $\Pi =$  “The graph has maximum degree of at least  $x$ .”
- $\Pi =$  “The graph has an  $h$ -index<sup>1</sup> [17] of at least  $x$ .”

Finally, we consider the dual parameterizations for hereditary graph properties characterized by a finite number of forbidden subgraphs. In the single-layer case, this problem has been studied by Lewis and Yannakakis [29] as well as Cai [8].

**Proposition 2.** Let  $\Pi$  be a hereditary graph property that is characterized by finitely many forbidden induced subgraphs. Then  $\Pi$ -ML-SUBGRAPH is NP-hard and FPT when parameterized by the number  $t - \ell$  of layers to delete and the number  $|V| - k$  of vertices to delete combined. It also admits a polynomial-size problem kernel with respect to these parameters.

<sup>1</sup>See Appendix A for a definition of  $h$ -index.

*Proof.* To see the fixed-parameter tractability, consider the search-tree algorithm that recursively searches for a forbidden induced subgraph  $G'$  in one of the layers, and branches, for each vertex  $v$  in  $G'$ , into the branch of deleting  $v$  and, additionally, into the branch of deleting the layer of  $G'$ . Finding  $G'$  takes polynomial time because it has constant size. Furthermore, each node in the resulting search tree has a constant number of children. Hence, the search-tree algorithm has a running time of  $c^{t-\ell+|V|-k} \cdot \text{poly}(I)$ , where  $c$  is a constant, and  $I$  is the instance size, as required.

To see that  $\Pi$ -ML-SUBGRAPH admits a polynomial kernel with respect to  $t - \ell$  and  $|V| - k$ , we use a (basically folklore) reduction to 2-COLOR HITTING SET, a variant of HITTING SET. Herein, we are given two disjoint ground sets  $B$  and  $W$ , a family  $\mathcal{F}$  of subsets of  $B \cup W$ , and two integers  $b, w$ . We are to decide whether there is a *hitting set*  $S \subseteq B \cup W$  containing  $b$  elements from  $B$  and  $w$  elements from  $W$ , that is, each subset  $F \in \mathcal{F}$  has  $F \cap S \neq \emptyset$ . Clearly, 2-COLOR HITTING SET is contained in NP.

The reduction works as follows. Given an instance of  $\Pi$ -ML-SUBGRAPH, we put the ground set  $B := V$  and put a distinct new vertex into  $W$  for each layer. For each layer, we enumerate all forbidden induced subgraphs. This takes polynomial time, as the maximum size of these subgraphs is a constant. To define  $\mathcal{F}$ , for each forbidden induced subgraph  $G'$  we add its vertex set  $V'$  plus the vertex  $v \in W$  corresponding to the layer in which  $G'$  is contained as a set  $V' \cup \{v\}$  to  $\mathcal{F}$ . Integer  $b$  is set to  $|V| - k$  and integer  $w$  to  $t - \ell$ . As mentioned, the reduction works in polynomial time. Since we have to “hit” each forbidden induced subgraph by either deleting a vertex from it, or deleting its layer completely, it is not hard to verify that the reduction is correct.

We now apply the so-called sunflower kernelization procedure [28, 33, 39] to the resulting 2-COLOR HITTING SET instance. A *sunflower* in  $\mathcal{F}$  is a subfamily  $\mathcal{F}' \subseteq \mathcal{F}$  such that there is a set  $C \subseteq B \cup W$  with the property that each pair  $F, F' \in \mathcal{F}'$  has  $F \cap F' = C$ . The *size* of a sunflower is  $|\mathcal{F}'|$ . If there is a sunflower of size  $b + w + 1$  in  $\mathcal{F}$ , then every hitting set contains at least one element of  $C$ . Hence, we can safely remove one set out of every sunflower of size at least  $b + w + 2$ . This can be done exhaustively in polynomial time [28, 33, 39]. After this procedure has been carried out, Erdős and Rado’s Sunflower Lemma [18] guarantees that the remaining set family  $\mathcal{F}$  has size  $O((b + w)^c)$ , where  $c$  is the size of the largest set in  $\mathcal{F}$ . This is a polynomial because the sets in  $\mathcal{F}$  have constant size. By removing elements of  $B \cup W$  which are not contained in any set in  $\mathcal{F}$ , we obtain an overall size bound on the resulting instance of 2-COLOR HITTING SET which is polynomial in  $b = |V| - k$  and  $w = t - \ell$ .

Finally, we transfer the instance of 2-COLOR HITTING SET created in this way to an equivalent instance of  $\Pi$ -ML-SUBGRAPH by using a polynomial-time many-one reduction. Such a reduction exists because 2-COLOR HITTING SET is in NP and  $\Pi$ -ML-SUBGRAPH is NP-hard for every graph property  $\Pi$  that is characterizable by a finite number of forbidden induced subgraphs [29].  $\square$

In the following corollary, we give a number of hereditary properties  $\Pi$  characterizable with a finite number of forbidden subgraphs and, hence, for which  $\Pi$ -ML-SUBGRAPH is fixed-parameter tractable with respect to the number  $t - \ell$  of layers to delete and the number  $|V| - k$  of vertices to delete combined. For their definitions see Appendix A.

**Corollary 3.**  *$\Pi$ -ML-SUBGRAPH is NP-hard and FPT when parameterized by  $t - \ell$  and  $|V| - k$  combined for  $\Pi \in \{\text{Cluster Graph}, \text{Cograph}, \text{Line Graph}, \text{Split Graph}\}$ .*

### 3 Non-hereditary Graph Properties

In this section, we give two results related to graph properties that are not necessarily hereditary. First, motivated by Connectivity, we give an FPT-algorithm with respect to  $t$  for graph properties in which each graph admits a certain nice vertex partitioning; this algorithm is also an XP-algorithm with respect to  $\ell$ . Second, we give a general W[1]-hardness reduction for the combined parameter  $k$  and  $\ell$ , capturing many classes of graph properties such as  $c$ -Core,  $c$ -Truss, Connectivity, and Matching.

**Vertex-partitionable graphs.** We start with investigating graph properties  $\Pi$  that allow for efficiently computable partitions of the graph into maximal components that each satisfy  $\Pi$ . It turns out that finding large  $\Pi$ -subgraphs in all input networks is tractable. This can be seen as a generalization of the component-detection algorithm in two layers by Gai et al. [21].

**Proposition 3.** *Let  $\Pi$  be a graph property such that for every graph  $G = (V, E)$  there is a partition  $\mathcal{P} := \{X_1, \dots, X_x\}$  of  $V$  such that:*

- $G[X_i] \in \Pi$  for all  $X_i \in \mathcal{P}$ ,
- for all  $X \subseteq V$  such that  $G[X] \in \Pi$ , we have  $X \subseteq X_i$  for some  $X_i \in \mathcal{P}$ , and
- $\mathcal{P}$  can be computed in  $T(|V|, |E|)$  time where  $T$  is non-decreasing in both arguments.

Then,  $\Pi$ -ML-SUBGRAPH can be solved in  $\binom{t}{\ell} \cdot O(n \cdot t) \cdot \max_{1 \leq i \leq t} (|E_i| + T(|V|, |E_i|))$  time.

We call a partition  $\mathcal{P}$  which fulfills the above conditions with respect to some graph property  $\Pi$  a  $\Pi$ -partition.

*Proof.* We describe an algorithm that outputs all maximal sets  $X \subseteq V$  such that  $G_i[X] \in \Pi$  for all input graphs  $G_i$ . We refer to these sets as *solutions* in the following. The algorithm maintains a partition  $\mathcal{P}$  of  $V$  where, initially  $\mathcal{P} = \{V\}$ .

The algorithm checks whether there is a  $Y \in \mathcal{P}$  such that  $G_i[Y] \notin \Pi$ . If this is the case, then it computes in  $T(|V|, |E_i|)$  time a  $\Pi$ -partition  $\mathcal{P}_Y$  of  $G_i[Y]$ . The partition  $\mathcal{P}$  is replaced by  $(\mathcal{P} \setminus \{Y\}) \cup \mathcal{P}_Y$ . If this is not the case, then the algorithm outputs all  $Y \in \mathcal{P}$ .

To see the correctness of the algorithm, first observe that for each output  $Y$ , we have  $G_i[Y] \in \Pi$  for all input graphs  $G_i$ . To show maximality of each  $Y$ , we show that the algorithm maintains the invariant that each solution  $X$  is a subset of some  $Y \in \mathcal{P}$ . This invariant is trivially fulfilled for the initial partition  $\{V\}$ . Now consider a set  $Y$  that is further partitioned by the algorithm. By the invariant, any solution  $X$  that has nonempty intersection with  $Y$  is a subset of  $Y$ . Furthermore, since  $\mathcal{P}_Y$  is a  $\Pi$ -partition of  $G_i[Y]$ , there is no solution  $X$  that contains vertices of two distinct sets  $Y_1, Y_2$  of  $\mathcal{P}_Y$ . Thus, each solution that is a subset of  $Y$  is also a subset of some  $Y' \in \mathcal{P}_Y$ . Hence, each output set  $X$  is a solution as it is an element of the final partition  $\mathcal{P}$  and all solutions are subsets of elements of  $\mathcal{P}$ .

To bound the running time, observe that for each  $Y \in \mathcal{P}$ , we can test in  $O(t \cdot \max_{1 \leq i \leq t} T(|V|, |E_i|))$  time whether it needs to be partitioned further. At most  $n$  partitioning steps are performed and if a set  $Y \in \mathcal{P}$  does not need to be partitioned further, then it can be discarded for the remainder of the algorithm. Thus, in  $O(n)$  applications of the “maximality test” the result is that  $Y$  is a solution and in  $O(n)$  applications of the maximality test,  $Y$  is further partitioned. Hence, the overall number of sets  $Y$  that are elements of  $\mathcal{P}$  at some point is  $O(n)$ . The overall running time now follows from the assumptions on  $T$  and from the fact that the induced subgraphs for all  $G_i$  can be computed in  $O(t \cdot \max_{1 \leq i \leq t} |E_i|)$  time for each  $Y$ .  $\square$

Examples of graph properties covered by Proposition 3 are **Connectivity**,  **$c$ -Connectivity**, and  **$c$ -Edge-Connectivity**. If we assume that graphs on one vertex are considered as (trivial)  $c$ -cores, then the  **$c$ -Core** property is covered: the nontrivial  $c$ -core of a graph is uniquely determined (it is the subgraph remaining after deleting any vertex with degree less than  $c$ ). Similarly, the  **$c$ -Truss** property is covered by Proposition 3 if we allow one-vertex graphs to be considered as  $c$ -trusses. Observe that we can easily choose to either incorporate or disregard connectivity from the  $c$ -core and  $c$ -truss definitions. For definitions of the graph properties mentioned above and in the following corollary see Appendix A.

If  $T$  is a polynomial function, which holds for all examples described above, then  $\Pi$ -ML-SUBGRAPH is fixed-parameter tractable with respect to  $t$  and polynomial time-solvable if  $\ell$  or  $t - \ell$  are constants.

**Corollary 4.**  $\Pi$ -ML-SUBGRAPH is FPT when parameterized by  $t$  and polynomial time-solvable if  $\ell$  or  $t - \ell$  are constants for  $\Pi \in \{\text{Connectivity}, c\text{-Connectivity}, c\text{-Edge-Connectivity}, c\text{-Truss}, c\text{-Core}\}$ .

**General hardness reduction.** Finally, we aim to give a general description of properties  $\Pi$  for which  $\Pi$ -ML-SUBGRAPH is NP-hard and W[1]-hard when parameterized by  $k$  and  $\ell$  combined. The next theorem is somewhat technical but covers many natural graph properties which are not covered by Proposition 1. Furthermore, it covers all graph properties from Corollary 4 and shows that for those properties  $\Pi$ -ML-SUBGRAPH becomes intractable when parameterized by  $\ell$  instead of  $t$ . We list some of them in Corollary 5.

**Theorem 1.** Let  $\Pi$  be a graph property.  $\Pi$ -ML-SUBGRAPH is W[1]-hard when parameterized by  $k$  and  $\ell$  combined, if there is an algorithm  $A$  that takes as input a vertex set  $W$ , a vertex set  $W' \subseteq W$  and an integer  $\alpha$  and computes a graph  $G = (V, E)$ , such that the following conditions hold.

- For each  $v \in W$  there is a vertex set  $X_v$  with  $|X_v| = f(\alpha)$  for some function  $f$ ,
- $\{X_v \mid v \in W\} \cup \{Y\}$  is a partition of  $V$  for some  $Y$  with  $|Y| = f'(\alpha)$  for some function  $f'$ ,

- for all  $X \subseteq V$  with  $|X| \geq \alpha \cdot f(\alpha) + f'(\alpha)$  we have that

$$G[X] \in \Pi \Leftrightarrow \exists W'' \subseteq W', \text{ such that } X = \bigcup_{v \in W''} X_v \cup Y,$$

- and  $A$  has running time  $f''(\alpha) \cdot |W|^{O(1)}$ , for some function  $f''$ .

If  $f''$  is polynomial, then we additionally get NP-hardness of  $\Pi$ -ML-SUBGRAPH.

The intuition is that each set  $X_v$  corresponds to one vertex  $v \in W$  and every set  $X$  such that  $G[X] \in \Pi$  either fully contains  $X_v$  or not. Furthermore,  $Y$  contains vertices that have to be included in  $X$  in order to have that  $G[X] \in \Pi$  and all sets  $X_v$  that correspond to vertices in  $v \in W \setminus W'$  have to be fully excluded from  $X$  in order to have that  $G[X] \in \Pi$ . For the proof, we reduce from BICLIQUE, which is W[1]-hard when parameterized by the size  $h$  of the biclique [30].

*Proof.* We give a parameterized reduction from BICLIQUE which, given an undirected graph  $H$  and a positive integer  $h$ , asks whether  $H$  contains a complete bipartite subgraph  $K_{h,h}$ . Let  $(H = (U, F), h)$  be an instance of BICLIQUE and let  $h \geq 2$ . We construct an instance of  $\Pi$ -ML-SUBGRAPH in the following way.

For all  $v \in U$ , let  $N_H(v)$  be the neighborhood of  $v$  with respect to  $H$ . Run Algorithm  $A$  on input  $(U, N_H(v), h)$  to create graphs  $G_v$  for each  $v \in U$ . Set  $k = h \cdot f(h) + f'(h)$  and  $\ell = h$ . Now we show that  $(\{G_v\}, k, \ell)$  is a yes-instance of  $\Pi$ -ML-SUBGRAPH if and only if  $(H, h)$  is a yes-instance of BICLIQUE.

$\Rightarrow$ : Assume that  $(H, h)$  is a yes-instance of BICLIQUE and let  $(C, D)$  with  $C, D \subseteq U$  and  $|C| = |D| = h$  be a biclique. Then for  $X = \bigcup_{v \in C} X_v \cup Y$  and  $v' \in D$ , we have that all  $v$ , such that  $X_v \subset X$ , are neighbors of  $v'$  and hence  $G_{v'}[X] \in \Pi$ . Note that  $|X| = h \cdot f(h) + f'(h)$  and  $|\{G_v \mid v \in D\}| = h$ , therefore it is a solution of  $\Pi$ -ML-SUBGRAPH.

$\Leftarrow$ : Assume that  $(\{G_v\}, k, \ell)$  is a yes-instance of  $\Pi$ -ML-SUBGRAPH, then we know that there are graphs  $G_i$ , with  $i \in L$ ,  $L \subseteq U$ ,  $|L| \geq h$ , and a vertex set  $X \subseteq V$  with  $|X| \geq k$ , such that  $G_i[X] \in \Pi$  for all  $i \in L$ . By the construction of  $G_i$ , we know that  $X = \bigcup_{v \in W'} X_v \cup Y$  for some  $W' \subseteq U$  with  $|W'| \geq h$ . Furthermore, we know that if  $i \in L$  then for all  $j \in W'$  (that is  $X_j \subset X$ ) we have that  $i$  is neighbor of  $j$ . Lastly, we have that  $i \in L$  implies that  $X_i \not\subset X$  and hence  $i \notin W'$ . Therefore we have that  $(L, W')$  is a biclique in  $H$  with  $|L| \geq h$  and  $|W'| \geq h$ .  $\square$

In the following corollary, we give several properties that are polynomial-time solvable in the single-layer case but NP-hard and W[1]-hard when parameterized by  $k$  and  $\ell$  combined in the multi-layer case. For their definitions see Appendix A.

**Corollary 5.**  $\Pi$ -ML-SUBGRAPH is NP-hard and W[1]-hard when parameterized by  $k$  and  $\ell$  combined for  $\Pi \in \{\text{Connectivity, Tree, Star, } c\text{-Core, } c\text{-Connectivity, } c\text{-Truss, Matching, } c\text{-Factor}\}$ .

*Proof Sketch.* We sketch Algorithm  $A$  from Theorem 1 for all properties listed above.

- **Connectivity, Tree, Star, 1-Core:** Let  $X_v := \{v\}$  and  $Y := \{u\}$ . Create an edge  $\{u, v\}$  for each vertex  $v \in W'$ .
- **$c$ -Core,  $c$ -Connectivity,  $c > 1$ :** Let  $X_v := \{v\}$  and  $Y := \{u_1, \dots, u_c\}$ . Create all edges  $(u, v)$  with  $u \in Y$  and  $v \in W'$ .
- **$c$ -Truss:** Let  $X_v := \{v\}$  and  $Y := \{u_1, \dots, u_{c+1}\}$ . Create all edges  $(u, v)$  with  $u \in Y$  and  $v \in W' \cup Y$  and  $u \neq v$ .
- **Matching:** Let  $X_v := \{v_1, v_2\}$  for each  $v \in W$  and create edge  $(v_1, v_2)$  if  $v \in W'$ .
- **$c$ -Factor:** For each  $v \in W'$ , add a connected  $c$ -regular graph of size  $f(c)$  to  $G$ , for each  $v \in W \setminus W'$ , add  $f(c)$  vertices to  $V$ , for some function  $f$ .  $\square$

A particular consequence of Corollary 5 is that the connected component detection algorithm for two layers by Gai et al. [21] does not generalize to Connectivity-ML-SUBGRAPH with  $\ell \ll t$  without significant running time overhead.

## 4 Matching and $c$ -Factors

In this section we consider the problem of finding a set  $X$  of at least  $k$  vertices that induces in  $\ell$  of  $t$  layers a subgraph that has a perfect matching. We also consider the more general  $c$ -factor property,

asking for a subset of edges such that each vertex is incident with exactly  $c$  edges. A perfect matching is a 1-factor. Finding  $c$ -factors is polynomial-time solvable for all  $c$  (see Plummer [36] for an overview on graph factors).

Corollary 5 states that **Matching-ML-SUBGRAPH** is  $W[1]$ -hard when parameterized by  $k$  and  $\ell$  combined. Through closer inspection we can get a stronger result. We show that **Matching-ML-SUBGRAPH** is polynomial-time solvable for  $\ell \leq 2$  and becomes  $W[1]$ -hard when parameterized by  $k$  already for  $\ell \geq 3$ . For **c-Factor-ML-SUBGRAPH** we show that it is already  $W[1]$ -hard when parameterized by  $k$  if  $\ell \geq 2$ .

For  $\ell = 1$ , we can simply check whether there is a  $c$ -factor in any of the layers. For  $\ell = 2$  **Matching-ML-SUBGRAPH** can be solved by reducing it to **MAXIMUM WEIGHT MATCHING**. To this end, let  $G_1 = (V, E_1)$  and  $G_2 = (V, E_2)$  be two input graphs for which we would like to know whether there is an  $X \subseteq V$  of size at least  $k$  such that both  $G_1[X]$  and  $G_2[X]$  have a perfect matching. We solve the problem by a simple reduction to **MAXIMUM WEIGHT MATCHING**, where we assume that the graph has edge weights and the task is to find a matching with maximum edge weights.

**Lemma 1.** *Given two graphs  $G_1 = (V, E_1)$  and  $G_2 = (V, E_2)$ , define a graph  $G' = (V', E')$  as follows:*

- $V' = \{v_1, v_2 \mid v \in V\}$  and
- $E' = \{\{v_1, v_2\} \mid v \in V\} \cup \{\{u_i, v_i\} \mid \{u, v\} \in E_i\}$ .

*Define a weight function  $w: E' \rightarrow \mathbb{N}$  as follows; let  $n := |V|$ :*

$$w(\{u_i, v_j\}) = \begin{cases} n & \text{if } i \neq j \text{ and } u = v, \\ n + 1 & \text{if } i = j \text{ (and } u \neq v). \end{cases}$$

*Let  $k \in \mathbb{N}$ . Then there is a set  $X \subseteq V$  of size at least  $k$  such that both  $G_1[X]$  and  $G_2[X]$  have a perfect matching if and only if the graph  $G'$  has a matching of  $w$ -weight at least  $n^2 + k$ .*

*Proof.* Assume that  $G'$  has a matching  $M' \subseteq E'$  with  $w(M') \geq n^2 + k$ . Let us first check that  $M'$  is in fact a perfect matching: If  $|M'| \leq n - 1$  then  $w(M') \leq (n + 1)(n - 1) = n^2 - 1 < n^2 + k$ , which would contradict the choice of  $M'$ . Thus  $|M'| \geq n$  but then  $M'$  must have exactly  $n$  edges since  $G'$  has  $2n$  vertices. That is,  $M'$  is a perfect matching. Let  $Y := \{v \mid v \in V \wedge \{v_1, v_2\} \in M'\}$ , that is,  $Y$  is the set of vertices of  $V$  whose copies in  $G'$  are matched under  $M'$ . Let  $X := V \setminus Y$ . It can be easily checked that both  $G_1[X]$  and  $G_2[X]$  have perfect matchings; we show this for  $G_1[X]$ : For any  $v \in X$  we know that  $\{v_1, v_2\} \notin M'$ , or else we would have  $v \in Y$  and  $v \notin X$ . Thus, using that  $M'$  is a perfect matching,  $v_1$  must be matched to another vertex  $u_1$ , which is then also in  $X$ , by definition. It follows that  $M'$  induces a perfect matching on  $G_1[X_1]$  where  $X_1 := \{v_1 \mid v \in X\}$ . Since  $G_1[X]$  is an isomorphic copy of  $G'[X]$  under canonical isomorphism  $\phi: v \mapsto v_1$ , we get that  $G_1[X]$  also has a perfect matching.

It remains to check that  $X$  has size at least  $k$ : Observe that  $|X| + |Y| = n$  since every vertex of  $V$  is either in  $X$  or in  $Y$ . Each  $v \in Y$  corresponds to a matching edge  $\{v_1, v_2\} \in M'$  which has weight  $n$  under  $w$ . Thus, if  $|X| < k$  then  $|Y| > n - k$ , which implies that  $w(M') \leq |X| \cdot (n + 1) + |Y|n < kn + k + (n - k)n = n^2 + k$ , contradicting the choice of  $M'$ .

Assume now that both  $G_1[X]$  and  $G_2[X]$  have perfect matchings  $M_1$  and  $M_2$  for some  $X$  of size at least  $k$ . Clearly, the size of  $X$  must be even. Define a matching  $M'$  of  $G'$  by  $M' := \{\{v_1, v_2\} \mid v \in V \setminus X\} \cup \{\{u_i, v_i\} \mid \{u, v\} \in M_i\}$ . In other words,  $M'$  is obtained by copying  $M_1$  and  $M_2$  to  $G'$  in the obvious way and matching all leftover vertices by the edges between the copies of the same vertex.

Clearly, for each vertex  $v \in V \setminus X$  this adds an edge  $\{v_1, v_2\}$  of weight  $n$  to  $M'$ . From  $M_1$  and  $M_2$  we copied  $\frac{|X|}{2}$  edges each, which results in exactly  $|X| \geq k$  edges of weight  $n + 1$  in  $M'$ . Thus, the total weight of  $M'$  is  $n^2 + k$ , as claimed.  $\square$

To show that **Matching-ML-SUBGRAPH** remains  $W[1]$ -hard when parameterized by  $k$  for  $\ell \geq 3$ , we reduce from **MULTICOLORED CLIQUE** which is known to be  $W[1]$ -hard when parameterized by the solution size [19].

**Theorem 2.** *Matching-ML-SUBGRAPH can be solved in polynomial time if  $\ell \leq 2$ . It is NP-hard and  $W[1]$ -hard when parameterized by  $k$  for all  $\ell \geq 3$  and  $t \geq \ell$ .*

*For  $c \geq 2$ , c-Factor-ML-SUBGRAPH is NP-hard and  $W[1]$ -hard when parameterized by  $k$  for all  $\ell \geq 2$  and  $t \geq \ell$ .*



*Proof.* The proof is split into two parts. First we prove the statement about **Matching-ML-SUBGRAPH** and then the statement about **c-Factor-ML-SUBGRAPH**.

**Matching.** We get the polynomial-time solvability of **Matching-ML-SUBGRAPH** for the case of  $\ell \leq 2$  from Lemma 1. For the case of  $\ell \geq 3$  we give a parameterized reduction from **MULTICOLORED CLIQUE**. This reduction can be adapted to the  $c$ -factor case.

In **MULTICOLORED CLIQUE**, we are given an  $h$ -partite graph  $H = (U_1 \uplus \dots \uplus U_h, F)$  and need to determine whether it contains a clique of size  $h$ . (Note that such a clique necessarily contains exactly one vertex from each set  $U_i$  and cliques of more than  $h$  vertices are impossible.) Without loss of generality, we assume that the number  $h$  of colors is even. We construct an instance of **Matching-ML-SUBGRAPH** for  $t = \ell = 3$  as follows and then argue that the construction is easily generalizable.

**Vertices.** First, create  $h - 1$  vertices for each vertex in graph  $H$  (one vertex for each color other than his own color). Formally, for each color  $1 \leq j \leq h$  and each  $u_i \in U_j$  create the vertex set  $V_i$  consisting of the vertices  $v_{(i,j')}$ ,  $j' \in (\{1, \dots, h\} \setminus \{j\})$ . Second, create one *color vertex*  $w_j$  for each color  $j \in \{1, \dots, h\}$ . We denote the set of color vertices as  $W := \bigcup_{1 \leq j \leq h} \{w_j\}$ .

**Vertex selection gadget by graph  $G_1$  and  $G_2$ .** For each color  $1 \leq j \leq h$  create for each  $u_i \in U_j$  one cycle on  $\{w_j\} \cup V_i$  in the graph  $G_1 \cup G_2$  such that the edges are alternating from  $G_1$  and from  $G_2$ . These  $|U_j|$  cycles are all of length  $h$  and share only color vertex  $w_j$ . To realize this, create the following edges. For each  $1 \leq z \leq h - 2$  create an edge in graph  $G_{(z \bmod 2) + 1}$  between  $v_{(i,z)}$  and  $v_{(i,z+1)}$  if  $z < j - 1$ , between  $v_{(i,z+1)}$  and  $v_{(i,z+2)}$  if  $z \geq j$ , and between  $v_{(i,z)}$  and  $v_{(i,z+2)}$  if  $z = j - 1$ . Create an edge between  $w_j$  and  $v_{(i,1)}$  in graph  $G_2$ , between  $v_{(i,h)}$  and  $w_j$  in graph  $G_1$  if  $j \neq h$ , and between  $v_{(i,h-1)}$  and  $w_j$  in graph  $G_1$  if  $j = h$ .

**Validation gadget by graph  $G_3$ .** For each adjacent vertex pair  $u_i, u_{i'}$  with  $u_i \in U_j$  and  $u_{i'} \in U_{j'}$ , we create an edge between  $v_{(i,j')}$  and  $v_{(i',j)}$  in  $G_3$ . Furthermore, create the edge  $\{w_j, w_{j+h/2}\}$  for each  $1 \leq j \leq h/2$ .

Finally, by setting  $k = h^2$  and  $t = \ell = 3$  we complete the construction, which can clearly be performed in polynomial time. It remains to show that graph  $H$  has a clique that contains each color exactly once if and only if there is a vertex set  $X \subseteq V$  with  $|X| \geq k$  such that graph  $G_z[X]$  contains a perfect matching for each  $1 \leq z \leq 3$ .

**Correctness.**  $\Rightarrow$ : Assume that graph  $H$  has a clique  $K := \{u_1, u_2, \dots, u_h\}$  and, without loss of generality,  $u_i \in U_i$  for all  $1 \leq i \leq h$ . We show that  $X := W \cup V_1 \cup V_2 \cup \dots \cup V_h$  is a solution for our **Matching-ML-SUBGRAPH** instance. By construction,  $X$  is of size  $k = h + h \cdot (h - 1) = h^2$ . It remains to show that graph  $G_z[X]$  has a perfect matching for each  $1 \leq z \leq 3$ . Recall that we created for each color  $1 \leq j \leq h$  and for each vertex  $u_i \in U_j$  one cycle on  $\{w_j\} \cup V_i$  in the graph  $G_1 \cup G_2$  such that the edges are alternating from  $G_1$  and from  $G_2$ . Since  $X$  only contains one of these cycles for each color, a perfect matching is easy to find for graph  $G_1[X]$  and for graph  $G_2[X]$ . For graph  $G_3[X]$ , we can find the matching  $\{\{v_{(i,j)}, v_{(j,i)}\} \mid 1 \leq i, j \leq h \wedge i \neq j\} \cup \{\{w_j, w_{j+h/2}\} \mid 1 \leq j \leq h/2\}$ , since, by construction,  $v_{(i,j)}$  is adjacent to  $v_{(j,i)}$  if  $u_i$  is adjacent to  $u_j$ , and  $u_i \in U_i$  and  $u_j \in U_j$  (which is the case since the vertices from  $K$  form a clique).

$\Leftarrow$ : Assume that there is a vertex set  $X \subseteq V$  with  $|X| \geq k$  such that graph  $G_z[X]$  contains a perfect matching for each  $1 \leq z \leq 3$ . First, consider the graph  $G_1 \cup G_2$  and some pair of vertices  $\{x_1, x_2\} \subseteq X$  that is matched in  $G_1$  or in  $G_2$ . Then, these two vertices must be from the same cycle  $\{w_j\} \cup V_i$  for some  $1 \leq j \leq h$  and  $u_i \in U_j$ , since otherwise there is no edge between them in any of the two graphs. Furthermore, if two vertices from  $\{w_j\} \cup V_i$  are in  $X$ , then all vertices from  $\{w_j\} \cup V_i$  must be in  $X$  because otherwise there is no matching either in  $G_1$  or in  $G_2$ : Every vertex except  $w_j$  has exactly one neighbor in  $G_1$  and one neighbor in  $G_2$  which are both enforced to be also contained in  $X$ —this enforces the whole cycle  $\{w_j\} \cup V_i$  to be contained in  $X$ . However,  $X$  contains the vertices from  $\{w_j\} \cup V_i$  for at most one  $i$  for every color  $1 \leq j \leq h$ , because  $w_j$  can only be matched to one vertex in  $G_1$  and to one vertex in  $G_2$ . This implies that  $X$  contains for each  $1 \leq j \leq h$  all vertices from  $\{w_j\} \cup V_i$  for exactly one  $i$ , since  $|X| \geq k = h^2$  and  $|\{w_j\} \cup V_i| = h$ . Without loss of generality, let  $V_i \subseteq X$  for all  $1 \leq i \leq h$  and let  $u_i$  be the vertex in graph  $H$  corresponding to  $V_i$ . We show that  $K = \{u_1, u_2, \dots, u_h\}$  is a clique in  $H$ . In graph  $G_3[X]$  each color vertex  $w_j$  must be matched to its only neighbor:  $w_{j+h/2}$  if  $j \leq h/2$  and  $w_{j-h/2}$  if  $j > h/2$  (and cannot be matched to  $v_{(i,j)}$ -vertices). Assume towards a contradiction that there is a pair of vertices  $\{u_i, u_{i'}\} \subseteq K$ , with  $u_i \in U_j$  and  $u_{i'} \in U_{j'}$ , that is not adjacent. Then, note that  $v_{(i,j')}$  and  $v_{(i',j)}$  must be matched, since, by construction of  $G_3$ , vertex  $v_{(i,j')}$  is only adjacent to vertex  $v_{(i',j)}$ . Moreover, if  $v_{(i,j')}$  is adjacent to  $v_{(i',j)}$  in  $G_3$ , then  $u_i$  is adjacent to  $u_{i'}$  in  $H$ —a contradiction.

Note that in order to make this reduction work for any  $t \geq \ell \geq 3$ , we can insert  $\ell - 3$  additional layers

of complete graphs and  $t - \ell$  layers of edgeless graphs.

**c-Factor.** In the following, we prove that for  $c \geq 2$ ,  $c$ -FACTOR-ML-SUBGRAPH is  $W[1]$ -hard when parameterized by  $k$  for  $t = \ell = 2$  and then argue that the construction is easily generalizable. Again, we give a parameterized reduction from MULTICOLORED CLIQUE. Here we are given an  $h$ -partite graph  $H = (U_1 \uplus \dots \uplus U_h, F)$  and need to determine whether it contains a clique of size  $h$ . (Note that such a clique necessarily contains exactly one vertex from each set  $U_i$  and cliques of more than  $h$  vertices are impossible.) We say that vertices from  $U_i$  have *color*  $i$ . Assume that  $h \geq c + 1$  and that  $c \cdot h$  is even.

Note that the idea of the reduction is similar to the one of **matching** but already works for  $\ell = 2$  in the  $c$ -factor case ( $c \geq 2$ ). One of the reasons is that we can construct connected  $c$ -regular graphs for  $c \geq 2$  of almost arbitrary size and this allows us to construct a vertex selection gadget with only one layer.

**Vertices.** For each color  $j$ ,  $1 \leq j \leq h$ , we do the following: We create a color vertex  $w_j$  and for each vertex  $u_i$  in  $U_j$ , we create a set of  $h - 1$  vertices  $V_i = \{v_{(i,j')}\mid 1 \leq j' \leq h \text{ and } j' \neq j\}$ . Note that we have one vertex in  $V_i$  for each color except the color of  $u_i$ . Let  $W = \{w_j \mid 1 \leq j \leq h\}$ . Furthermore, for each edge  $f \in F$  we create a set of vertices  $V_f$  with  $|V_f| = c - 1$ . Let  $V_F = \bigcup_{f \in F} V_f$  and  $V = \bigcup_i V_i \cup W \cup V_F$ .

**Vertex selection gadget by graph  $G_1$ .** For every  $1 \leq j \leq h$  and every  $u_i$  in  $U_j$  we do the following: We create a connected  $c$ -regular graph on the vertex set  $V_i \cup \{w_j\}$ . Note that this can be done in the following way: Order the vertices in  $V_i \cup \{w_j\}$  arbitrarily and connect each vertex to the  $\lfloor c/2 \rfloor$  subsequent vertices, wrapping around at the end. If  $c$  is odd, additionally connect each vertex  $v$  with the vertex at position  $(x + h/2) \bmod h$  in the ordering, where  $x$  is the position of vertex  $v$ . Furthermore, we create a complete graph on the vertices in  $V_F$ .

**Validation gadget by graph  $G_2$ .** For all edges  $f \in F$  we do the following: Let  $f$  be the edge between  $u_i$  and  $u_{i'}$ , and  $u_i \in U_j$  and  $u_{i'} \in U_{j'}$ . We create a complete graph on the vertices in  $V_f \cup \{v_{(i,j')}, v_{(i',j)}\}$ . Note that this complete graph has order  $c + 1$  and hence is a  $c$ -regular graph. Furthermore, we create a complete graph on all vertices in  $W$ .

By setting  $k = h^2 + \frac{1}{2}h(h-1) \cdot (c-1)$  we complete the construction, which can clearly be performed in polynomial time.

**Correctness.**  $\Rightarrow$ : Assume graph  $H$  has an  $h$ -colored clique  $K$  and without loss of generality  $K = \{u_1, u_2, \dots, u_h\}$ . Let  $F_K$  denote the set of all edges in the clique  $K$ . Furthermore, let  $V_K = \bigcup_{1 \leq i \leq h} V_i$  and  $V_{F_K} = \bigcup_{f \in F_K} V_f$ . We show that  $X = W \cup V_K \cup V_{F_K}$  is a solution for  $c$ -FACTOR-ML-SUBGRAPH. Note that by construction,  $|X| = h^2 + \frac{1}{2}h(h-1) \cdot (c-1)$ . It remains to show that  $G_1[X]$  and  $G_2[X]$  each have  $c$ -factors. Observe that for any graph  $G = (V, E)$  and any partition  $\mathcal{P} = \{P_1, P_2, \dots, P_p\}$  of  $V$  we have that if  $G[P_i]$  has a  $c$ -factor for all  $i$ ,  $1 \leq i \leq p$ , then  $G$  has a  $c$ -factor as well.

1. Note that  $G_1[V_{F_K}]$  is a complete graph of order  $\frac{1}{2}h(h-1) \cdot (c-1) \geq 2c + 1$  and hence has a  $c$ -factor. Furthermore, let  $u_i \in K$  and assume that  $u_i \in U_j$ . Then we have that  $G_1[V_i \cup \{w_j\}]$  is by construction a  $c$ -regular graph and hence also has a  $c$ -factor. Since  $K$  is an  $h$ -colored clique,  $\{V_i \cup \{w_j\} \mid u_i \in K \text{ and } u_i \in U_j\}$  is a partition of  $V_K \cup W$ .
2. Note that  $G_2[W]$  is a complete graph of size  $h \geq c + 1$  and hence has a  $c$ -factor. Let  $f \in F_K$  be the edge connecting  $u_i$  and  $u_{i'}$ , and let  $u_i \in U_j$  and  $u_{i'} \in U_{j'}$ . Then we have that  $G_2[V_f \cup \{v_{(i,j')}, v_{(i',j)}\}]$  is by construction a complete graph of order  $c + 1$  and hence also has a  $c$ -factor. Since  $K$  is an  $h$ -colored clique,  $\{V_f \cup \{v_{(i,j')}, v_{(i',j)}\} \mid f \in F_K \text{ and } f = \{u_i, u_{i'}\}\}$  is a partition of  $V_{F_K} \cup V_K$ . Notably,  $\mathcal{P}_K = \{\{v_{(i,j')}, v_{(i',j)}\} \mid \{u_i, u_{i'}\} = f \text{ for some } f \in F_K\}$  is a partition of  $V_K$ , since any  $u_i$  is connected to  $h-1$  other vertices with a different color each and therefore  $\{v_{(i,j')} \mid \{v_{(i,j')}, v_{(i',j)}\} \in \mathcal{P}_K\} = V_i$ .

$\Leftarrow$ : Assume there is a vertex set  $X \subseteq V$  such that  $|X| \geq k$  and  $G_i[X]$  has a  $c$ -factor for  $\ell$  different layers  $i$ . By construction of the layers we have that  $G_1[X]$  and  $G_2[X]$  both have  $c$ -factors. Furthermore, we show the following facts:

**Fact 1:** If  $v_{(i,j')} \in X$  and  $u_i \in U_j$  then  $V_i \subseteq X$  and  $w_j \in X$ : By construction  $G_1[V_i \cup \{w_j\}]$  is a connected  $c$ -regular graph and each  $v_{(i,j')} \in V_i$  is only connected to other vertices in  $V_i \cup \{w_j\}$  in  $G_1$ . Notice that any proper subgraph of a connected  $c$ -regular graph is not  $c$ -regular and does not have a  $c$ -factor. It follows that as soon as any  $v_{(i,j')} \in V_i$  is included in  $X$ , all other vertices in  $V_i$  have to be included as well as  $w_j$ , the vertex corresponding to the color of  $u_i$ .

**Fact 2:** If  $V_i \subseteq X$  and  $V_{i'} \subseteq X$  and  $u_i \in U_j$ , then  $u_{i'} \notin U_{j'}$ : Note that  $G_1[V_{i'}]$  does not have a  $c$ -factor: By construction we get a connected  $c$ -regular graph by adding the vertex corresponding to the color of  $u_{i'}$ , hence  $G_1[V_{i'}]$  is a proper subgraph of a connected  $c$ -regular graph. Furthermore, we

have that  $w_j$  is already part of the  $c$ -regular spanning graph of  $G_1[V_i \cup \{w_j\}]$ , therefore  $u_{i'}$  cannot have color  $j$ , that is  $u_{i'} \notin U_j$ .

**Fact 3:** If  $X \cap V_f \neq \emptyset$  for some  $f \in F$ , then  $V_f \cup \{v_{(i,j')}, v_{(i',j)}\} \subseteq X$ , where  $u_i \in U_j$  and  $u_{i'} \in U_{j'}$  are the endpoints of  $f$ : By construction  $G_2[V_f \cup \{v_{(i,j')}, v_{(i',j)}\}]$  is a clique of size  $c + 1$  and hence a connected  $c$ -regular graph. Furthermore, this clique is disconnected from the rest of  $G_2$  and hence as soon as one of its vertices is included in  $X$ , all of them are.

Now we show that  $K = \{u_i \mid V_i \subseteq X\}$  is an  $h$ -colored clique in  $H$ . First, we show that there must be some  $v_{(i,j)} \in X$ : Since  $|X| \geq h^2 + \frac{1}{2}h(h-1) \cdot (c-1)$  we have that  $X \neq W$  and from Fact 4 we get that  $X \not\subseteq W \cup V_F$ .

By Fact 4 we know that any  $V_i$  is either a subset of  $X$  or  $X \cap V_i = \emptyset$ . Furthermore, if  $V_i \subseteq X$  and  $u_i \in U_j$ , so is  $w_j$ . Fact 4 yields that we cannot have two vertices of the same color in  $K$ . This yields the following inequality.

$$|\bigcup_{V_i \subseteq X} V_i \cup W| \leq h^2 \quad (1)$$

Note that  $|X| \geq h^2 + \frac{1}{2}h(h-1) \cdot (c-1)$  and Inequality 1 imply that  $X \cap V_F \neq \emptyset$ . By Fact 4 we get that  $X$  can only include vertices corresponding to edges between vertices  $u_i$  and  $u_{i'}$ , such that  $V_i \cup V_{i'} \subseteq X$ . Therefore we get the following inequality.

$$|X \cap V_F| \leq \frac{1}{2}h(h-1) \cdot (c-1) \quad (2)$$

Note that we get  $|X| = h^2 + \frac{1}{2}h(h-1) \cdot (c-1)$  if and only if both Inequalities 1 and 2 are equalities. This implies that we have  $|K| = h$ , all vertices in  $K$  have different colors and all colors are present (Inequality 1), and  $K$  is a clique in  $H$  (Inequality 2).

Note that in order to make this reduction work for any  $t \geq \ell \geq 2$ , we can insert  $\ell - 2$  additional layers of complete graphs and  $t - \ell$  layers of edgeless graphs.  $\square$

## 5 Hamiltonian Paths

In this section we investigate the problem variant of finding Hamiltonian subgraphs. Note that this property is not captured by any of the meta theorems in Section 3. **HAMILTONIAN-SUBGRAPH** is known to be FPT when parameterized by the size of the subgraph  $k$  [32]. For the multi-layer case, we can show that it is already intractable for any  $\ell \geq 2$ .

**Theorem 3.** *HAMILTONIAN-ML-SUBGRAPH is NP-hard and W[1]-hard when parameterized by  $k$  for all  $\ell \geq 2$  and  $t \geq \ell$ .*

We reduce from the MULTICOLORED BICLIQUE problem. A simple reduction from CLIQUE shows that MULTICOLORED BICLIQUE is W[1]-hard [12].

*Proof.* In MULTICOLORED BICLIQUE we are given a bipartite graph  $H = (U \cup W, F)$  and a coloring that partitions vertices in  $U$  and  $W$  in  $h$  parts each, that is,  $U = U_1 \uplus \dots \uplus U_h$  and  $W = W_1 \uplus \dots \uplus W_h$ . We call these parts *colors*. We need to determine whether  $H$  contains a biclique of size  $2h$  that consists of one vertex of each color. Observe that the vertex-coloring implies that any solution contains  $h$  vertices from  $U$  and  $h$  vertices from  $W$ . We will call the vertices from  $U$  *low*, and the vertices from  $W$  *high*. Similarly, we call colors  $U_i$  *low* and colors  $W_i$  *high*. Given an instance  $H$  of MULTICOLORED BICLIQUE, we construct an instance of HAMILTONIAN-ML-SUBGRAPH for  $t = \ell = 2$  as follows and then argue that the construction is easily generalizable.

*Vertices.* The vertex set  $V$  consists of the following subsets:

- All vertices  $U \cup W$  of  $H$ ,
- $\{s_1, s_2\}$ , where we assume that  $s_1$  and  $s_2$  are not vertices of  $H$ ,
- $A_{i,j}$ ,  $1 \leq i \leq h$ ,  $1 \leq j \leq h$ , where  $A_{i,j} := \{\alpha_{\{u,w\}} \mid \{u,w\} \in F \wedge u \in U_i \wedge w \in W_j\}$ , and
- $D_{i,j}$ ,  $1 \leq i \leq h$ ,  $1 \leq j \leq h$ , where  $D_{i,j} := \{\delta_{\{w,u\}} \mid \{w,u\} \in F \wedge w \in W_i \wedge u \in U_j\}$ .

Informally, the latter two sets are constructed by adding two vertices for each edge of  $H$ , each corresponding to one orientation of the undirected edge. The vertices are then assigned to the vertex sets according

to the colors of their endpoints and the orientations. Oriented edges from  $U$  to  $W$  (and their corresponding vertices in  $V$ ) are called *ascending*, oriented edges from  $W$  (and their corresponding vertices) are called *descending*.

Now we describe how to construct a vertex and edge selection gadget by graph  $G_1$  and a validation gadget by graph  $G_2$ . Both graphs consist of  $2h^2 + 2h + 2$  levels with edges only between neighboring levels; each  $U_i$ ,  $W_i$ ,  $A_{i,j}$  and each  $D_{i,j}$  forms one level, the two remaining levels contain  $s_1$  and  $s_2$ , respectively.

*Vertex and edge selection gadget by graph  $G_1$ .* Informally, the graph  $G_1$  is constructed by putting all low vertices and their incident ascending edges into low levels, then adding two levels for  $s_1$  and  $s_2$  and then putting all high vertices and their incident descending edges into high levels. More precisely,  $U_1$  is the first level of  $G$  and  $A_{1,1}$  is the second level. Then edges are added from each  $u \in U_1$  to all vertices of  $A_{1,1}$  that correspond to an edge incident with  $u$ . Then, the ascending vertices “incident” with vertices from  $U_1$  are added for increasing colors of the high endpoints. Afterwards, the vertices from  $U_2$  are added and then the vertices corresponding to their incident edges, and so on. The special vertices  $s_1$  and  $s_2$  are added in two middle levels, separating the levels containing low vertices from those containing high vertices. Formally, the graph  $G_1$  is constructed as follows:

- For each  $u \in U_i$ ,  $1 \leq i \leq h$  add the edge  $\{u, \alpha_{\{u,w\}}\}$  if  $w \in W_1$ ,
- for each  $\alpha_{\{u,w\}} \in A_{i,h}$ ,  $1 \leq i < h$ , add an edge to each  $u' \in U_{i+1}$ ,
- for each  $\alpha_{\{u,w\}} \in A_{i,j}$ ,  $1 < j < h$ , add an edge to each  $\alpha_{\{u,w'\}} \in A_{i,j+1}$ ,
- for each  $\alpha_{\{u,w\}} \in A_{h,h}$  add an edge to  $s_1$ ,
- add the edge  $\{s_1, s_2\}$ ,
- for each  $w \in W_1$ , add the edge  $\{s_2, w\}$ ,
- for each  $w \in W_i$ ,  $1 < i \leq h$ , add the edge  $\{w, \delta_{\{w,u\}}\}$  if  $u \in U_1$ ,
- for each  $\delta_{\{w,u\}} \in D_{i,h}$ ,  $1 \leq i < h$ , add an edge to each  $w' \in W_{i+1}$ , and
- for each  $\delta_{\{w,u\}} \in D_{i,j}$ ,  $1 \leq j < h$ , add an edge to each  $\delta_{\{w,u'\}} \in D_{i,j+1}$ .

The idea behind the construction is that any path from the first level to the last level corresponds to a selection of  $2h$  vertices and of  $2h^2$  edges incident with these vertices.

*Validation gadget by graph  $G_2$ .* With the second graph  $G_2$  we enforce that the selected ascending and descending edges between each color pair  $U_i$  and  $W_j$  correspond to the same edge in  $H$  and that any path of length  $2h + 2h^2$  passes through each level of  $G_1$  and each level of  $G_2$ . Formally,  $G_2$  is constructed as follows, herein, assume an arbitrary but fixed ordering on pairs of low and high colors.

- Level 1 contains  $s_1$ ,
- level  $2i$ ,  $1 \leq i \leq h^2$ , contains all ascending vertices of the  $i$ -th color pair,
- level  $2i + 1$ ,  $1 \leq i \leq h^2$ , contains all descending vertices of the  $i$ -th color pair,
- level  $2h^2 + 2$  contains  $s_2$ ,
- level  $2h^2 + 2 + i$ ,  $1 \leq i \leq h$ , contains the vertices from  $U_i$ , and
- level  $2h^2 + h + 2 + i$ ,  $1 \leq i \leq h$ , contains the vertices from  $W_i$ .

All edges between consecutive levels are added except for the levels  $2i$  and  $2i + 1$ ,  $1 \leq i \leq h^2$ : here we add only an edge between vertices that correspond to the same edge, that is, we add the edge set  $\{\alpha_{\{u,w\}}, \delta_{\{w,u\}} \mid \{u,w\} \in F\}$ .

To finish the construction we set  $k = 2h + 2h^2 + 2$ . The reduction runs clearly in polynomial time and the new parameter  $k$  depends only on the parameter  $h$  of the MULTICOLORED BICLIQUE instance. Thus it remains to show equivalence of the instances. *Correctness.*  $\Rightarrow$ : Let  $K = \{u_1, \dots, u_h, w_1, \dots, w_h\}$  be a multi-colored biclique in  $H$ . Let  $X$  be the vertex set containing  $K$ ,  $s_1$  and  $s_2$ , and for each edge  $e$  of  $H[K]$ , the ascending and the descending vertex corresponding to  $e$ . We show that  $G_1[X]$  and  $G_2[X]$  have a Hamiltonian path. In  $G_1$ , this path starts at  $u_1$ , then visits  $\alpha_{u_1, w_1}$ , then  $\alpha_{u_1, w_2}$  until  $\alpha_{u_1, w_h}$ . Then it visits  $u_2$  and the ascending vertices corresponding to edges incident with  $u_2$  in the same fashion, that is, first  $\alpha_{u_2, w_1}$ , then  $\alpha_{u_2, w_2}$  and so on. This is continued until  $\alpha_{u_h, w_h}$  is visited. Then, the path visits  $s_1$  and  $s_2$ . Then it visits the high vertices and the descending vertices for their incident edges in the same fashion. By construction, all the necessary edges are present: neighboring edge vertices correspond to edges that share one endpoint, and after a vertex  $\alpha_i$ , the next visited edge vertex is incident with  $\alpha_i$ .

In  $G_2$ , the path visits each level exactly once, going from level 1 through level  $k$ . The necessary edges are present since the only neighboring levels that are not complete bipartite graphs are those that contain ascending and descending vertices of the same color pair. Since for each color pair the ascending and descending vertex in  $S$  correspond to the same edge in  $H$ , they are adjacent in  $G_2$ .

$\Leftarrow$ : Observe that any vertex set consisting only of vertices from  $\{s_1, s_2\} \cup \bigcup_{1 \leq i \leq h} (U_i \cup W_i)$  is disconnected either in  $G_1$  or in  $G_2$ . Thus, the set  $X$  contains at least one ascending or descending vertex. In

either case it must also contain an edge vertex of the other type: in  $G_2$ , the levels containing ascending and descending vertices alternate and any set containing either only vertices of the first two levels of  $G_2$  or only of vertices of the last  $2h+2$  levels is disconnected in  $G_1$ . Now since  $X$  contains an ascending and a descending vertex and since  $G_1[X]$  is connected, the vertices  $s_1$  and  $s_2$  are contained in  $X$ . Thus, the vertex set  $X$  contains a vertex from the first and the last level of  $2h+2h^2+2$  levels in  $G_2$ . This implies that it contains exactly one vertex of each level of  $G_2$ . Thus, the vertex set  $X$  contains exactly  $h$  low vertices and  $h$  high vertices with different colors and it contains for each color pair an ascending and a descending vertex. By construction of  $G_2$  and since  $G_2[X]$  has a Hamiltonian path visiting each level exactly once, these two vertices are the same, that is, the  $2h^2$  ascending and descending vertices correspond to  $h^2$  edges in  $H$ . By construction of  $G_1$  and since  $G_1[X]$  has a Hamiltonian path visiting each level of  $G_1$  exactly once, these edges are incident only with vertices of  $(U \cup W) \cap X$ . Thus,  $H[(U \cup W) \cap X]$  is a multicolored biclique.

Note that in order to make this reduction work for any  $t \geq \ell \geq 2$ , we can insert  $\ell - 2$  additional layers of complete graphs and  $t - \ell$  layers of edgeless graphs.  $\square$

## 6 Conclusion

We have initiated a systematic study of subgraph detection problems in multi-layer networks. In particular, we have shown hardness results for many multi-layer subgraph detection problems that are solvable in polynomial time in the single-layer case. In the following, we list some possibilities for obtaining positive algorithmic results. For example, the case of two-layer graphs should receive special attention. We showed that **Matching-ML-SUBGRAPH** is solvable in polynomial time in the two-layer case, whereas it is **W[1]**-hard when parameterized by  $k$  for 3 or more layers. Considering cycle-free subgraphs Agrawal et al. [1] also showed specialized algorithms for the two-layer case. Thus, it would be interesting to systematically determine which subgraph detection problems become tractable in the two-layer case and to identify problems that behave differently for two and three layers. Finally, in many applications the input graphs are directed. One of our hardness results transfers directly to this case: The construction in the reduction from **MULTICOLORED BICLIQUE** to **Hamiltonian-ML-SUBGRAPH** (proof of Theorem 3) can be easily adapted to yield directed acyclic graphs by orienting all edges from lower levels to higher levels. Hence, for directed acyclic graphs the complexity gap between the cases with one and two layers is even bigger because finding a longest path in a directed acyclic graph is polynomial-time solvable.

## References

- [1] A. Agrawal, D. Lokshtanov, A. E. Mouawad, and S. Saurabh. Simultaneous feedback vertex set: A parameterized perspective. In *Proceedings of the 33rd International Symposium on Theoretical Aspects of Computer Science (STACS '16)*, 2016. To appear, available at <http://arxiv.org/abs/1510.01557>.
- [2] J. Bang-Jensen and G. Gutin. *Digraphs: Theory, Algorithms and Applications*. Springer, 2nd edition, 2002.
- [3] M. Berlingerio, M. Coscia, F. Giannotti, A. Monreale, and D. Pedreschi. Multidimensional networks: foundations of structural analysis. *Proceedings of the 22nd International Conference on World Wide Web (WWW '15)*, 16(5-6):567–593, 2013.
- [4] S. Boccaletti, G. Bianconi, R. Criado, C. I. del Genio, J. Gmez-Gardees, M. Romance, I. Sendia-Nadal, Z. Wang, and M. Zanin. The structure and dynamics of multilayer networks. *Physics Reports*, 544(1):1–122, 2014.
- [5] B. Boden, S. Günnemann, H. Hoffmann, and T. Seidl. Mining coherent subgraphs in multi-layer graphs with edge labels. In *The 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '12)*, pages 1258–1266, 2012.
- [6] A. Brandstädt, V. B. Le, and J. P. Spinrad. *Graph Classes: A Survey*, volume 3 of *SIAM Monographs on Discrete Mathematics and Applications*. SIAM, 1999.
- [7] B. Bui-Xuan, M. Habib, and C. Paul. Competitive graph searches. *Theoretical Computer Science*, 393(1-3):72–80, 2008.
- [8] L. Cai. Fixed-parameter tractability of graph modification problems for hereditary properties. *Information Processing Letters*, 58(4):171–176, 1996.

- [9] L. Cai and J. Ye. Dual connectedness of edge-bicolored graphs and beyond. In *Proceedings of the 39th International Symposium on Mathematical Foundations of Computer Science (MFCS '14)*, volume 8635 of *LNCS*, pages 141–152. Springer, 2014.
- [10] J. Cohen. Trusses: Cohesive subgraphs for social network analysis. *National Security Agency Technical Report*, page 16, 2008.
- [11] M. Cygan, F. V. Fomin, L. Kowalik, D. Lokshtanov, D. Marx, M. Pilipczuk, M. Pilipczuk, and S. Saurabh. *Parameterized Algorithms*. Springer, 2015.
- [12] H. Dell and D. Marx. Kernelization of packing problems. In *Proceedings of the Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms (SODA '12)*, pages 68–81. SIAM, 2012.
- [13] R. Diestel. *Graph Theory*, volume 173 of *Graduate Texts in Mathematics*. Springer, 3rd edition, 2005.
- [14] X. Dong, P. Frossard, P. Vandergheynst, and N. Nefedov. Clustering with multi-layer graphs: A spectral perspective. *IEEE Transactions on Signal Processing*, 60(11):5820–5831, 2012.
- [15] X. Dong, P. Frossard, P. Vandergheynst, and N. Nefedov. Clustering on multi-layer graphs via subspace analysis on Grassmann manifolds. *IEEE Transactions on Signal Processing*, 62(4):905–918, 2014.
- [16] R. G. Downey and M. R. Fellows. *Fundamentals of Parameterized Complexity*. Springer, 2013.
- [17] D. Eppstein and E. S. Spiro. The  $h$ -index of a graph and its application to dynamic subgraph statistics. *Journal of Graph Algorithms and Applications*, 16(2):543–567, 2012.
- [18] P. Erdős and R. Rado. Intersection theorems for systems of sets. *Journal of the London Mathematical Society*, 35(1):85–90, 1960.
- [19] M. R. Fellows, D. Hermelin, F. A. Rosamond, and S. Vialette. On the parameterized complexity of multiple-interval graph problems. *Theoretical Computer Science*, 410(1):53–61, 2009.
- [20] J. Flum and M. Grohe. *Parameterized Complexity Theory*. Springer, 2006.
- [21] A.-T. Gai, M. Habib, C. Paul, and M. Raffinot. Identifying common connected components of graphs. Technical report, Technical Report RR-LIRMM-03016, LIRMM, Université de Montpellier II, 2003.
- [22] M. C. Golumbic. *Algorithmic graph theory and perfect graphs*, volume 57 of *Annals of Discrete Mathematics*. Elsevier B. V., 2nd edition, 2004. First edition Academic Press, 1980.
- [23] D. Jiang and J. Pei. Mining frequent cross-graph quasi-cliques. *ACM Transactions on Knowledge Discovery from Data*, 2(4), 2009.
- [24] M. Kano and X. Li. Monochromatic and Heterochromatic Subgraphs in Edge-Colored Graphs - A Survey. *Graphs and Combinatorics*, 24(4):237–263, 2008.
- [25] S. Khot and V. Raman. Parameterized complexity of finding subgraphs with hereditary properties. *Theoretical Computer Science*, 289(2):997–1008, 2002.
- [26] J. Kim and J. Lee. Community detection in multi-layer graphs: A survey. *SIGMOD Record*, 44(3):37–48, 2015.
- [27] M. Kivel, A. Arenas, M. Barthélemy, J. P. Gleeson, Y. Moreno, and M. A. Porter. Multilayer networks. *Journal of Complex Networks*, 2(3):203–271, 2014.
- [28] S. Kratsch. Polynomial kernelizations for  $\text{MIN } F^+_{\Pi_1}$  and  $\text{MAX NP}$ . In *Proceedings of the 26th International Symposium on Theoretical Aspects of Computer Science (STACS '09)*, pages 601–612. IBFI Dagstuhl, Germany, 2009.
- [29] J. M. Lewis and M. Yannakakis. The node-deletion problem for hereditary properties is NP-complete. *Journal of Computer and System Sciences*, 20(2):219–230, 1980.
- [30] B. Lin. The parameterized complexity of  $k$ -biclique. In *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA '15)*, pages 605–615. SIAM, 2015.
- [31] M. Magnani and L. Rossi. The ML-model for multi-layer social networks. In *Proceedings of the International Conference on Advances in Social Networks Analysis and Mining (ASONAM '11)*, pages 5–12. IEEE Computer Society, 2011.
- [32] B. Monien. How to find long paths efficiently. *North-Holland Mathematics Studies*, 109:239–254, 1985.
- [33] H. Moser. *Finding Optimal Solutions for Covering and Matching Problems*. PhD thesis, Institut für Informatik, Friedrich-Schiller Universität Jena, 2009.
- [34] P. J. Mucha, T. Richardson, K. Macon, M. A. Porter, and J.-P. Onnela. Community structure in time-dependent, multiscale, and multiplex networks. *Science*, 328(5980):876–878, 2010.
- [35] R. Niedermeier. *Invitation to Fixed-Parameter Algorithms*. Oxford University Press, 2006.

- [36] M. D. Plummer. Graph factors and factorization: 1985-2003: A survey. *Discrete Mathematics*, 307(7-8):791–821, 2007.
- [37] L. Rossi, M. Musolesi, and A. Torsello. On the  $k$ -anonymization of time-varying and multi-layer social graphs. In *Proceedings of the 9th International Conference on Web and Social Media (ICWSM '15)*, pages 377–386. AAAI Press, 2015.
- [38] S. B. Seidman. Network structure and minimum degree. *Social Networks*, 5(3):269–287, 1983.
- [39] R. van Bevern. Towards optimal and expressive kernelization for  $d$ -hitting set. *Algorithmica*, 70(1):129–147, 2014.
- [40] Z. Zeng, J. Wang, L. Zhou, and G. Karypis. Out-of-core coherent closed quasi-clique mining from large dense graph databases. *ACM Transactions on Database Systems*, 32(2):13, 2007.

## A Definitions

**Asteroidal Triple Free Graph** An independent set of size three where each pair of vertices is joined by a path that avoids the neighborhood of the third is called an *asteroidal triple*. A graph is *asteroidal triple-free* if it does not contain asteroidal triples.

**$c$ -Colorable Graph** A graph is *c-colorable* if there is a way of coloring the vertices with at most  $c$  different colors such that no two adjacent vertices share the same color.

**$c$ -Connectivity** A graph is called *c-connected* if it contains at least  $c + 1$  vertices, but does not contain a set of  $c - 1$  vertices whose removal disconnects the graph.

**$c$ -Core** A graph is called a *c-core* if each vertex has degree at least  $c$ .

**$c$ -Edge-Connectivity** A graph is called *c-edge-connected* if it does not contain a set of  $c - 1$  edges whose removal disconnects the graph.

**$c$ -Factor** A graph has a *c-factor* if it has a  $c$ -regular spanning graph.

**$c$ -Regular Graph** A graph is called *c-regular* if every vertex has degree  $c$ .

**$c$ -Truss** A graph is called a *c-truss* if each edge is contained in at least  $c - 2$  triangles.

**Chordal Graph** A graph is called *chordal* if each induced cycle has at most three vertices.

**Cluster Graph** A graph is called a *cluster graph* if it is a collection of cliques.

**Cograph** A graph is called a *cograph* if it does not contain any induced path of length four.

**Comparability Graph** A graph is called a *comparability graph* if there is a partial order over the vertices such that each pair of vertices is adjacent if and only if it is comparable.

**Complete Multipartite Graph** A graph is called *complete multipartite* if the vertex set can be partitioned such that each vertex pair is adjacent if and only if the two vertices are in different partitions.

**Edgeless Graph** A graph is called *edgeless* if it does not contain any edges.

**Forest** A graph is called a *forest* if it is a collection of trees.

**Hamiltonian Graph** A graph is called *Hamiltonian* if it contains a *Hamiltonian path*, that is a path that visits each vertex exactly once.

**$h$ -Index** The *h-index* of a graph is the largest integer  $h$  such that the graph contains at least  $h$  vertices with degree at least  $h$ .

**Interval Graph** A graph is called a *interval graph* if an interval of the real numbers can be assigned to each vertex such that two vertices are adjacent if and only if the intervals overlap.

**Line Graph** A graph is called a *line graph* if there is another graph such that each vertex of the line graph corresponds to an edge of the other graph and two vertices in the line graph are adjacent if the corresponding edges in the other graph share a common endpoint.

**Matching** A graph has a *matching* if it has an 1-factor.

**Perfect Graph** A graph is called *perfect* if the chromatic number of every induced subgraph equals the size of the largest clique of that subgraph. The *chromatic number* of a graph is the smallest  $c$  such that the graph is  $c$ -colorable.

**Permutation Graph** A graph is called a *permutation graph* if there is a permutation of the vertices such that two vertices are adjacent if and only if they are reversed by the permutation.

**Planar Graph** A graph is called *planar* if it can be embedded in the plane, that is it can be drawn on the plane such that its edges only intersect at their endpoints.

**Split Graph** A graph is called a *split graph* if its vertices can be partitioned into a clique and an independent set.

**Star** A graph is called a *star* if it is a tree with only one internal node.